|  |
| --- |
|  |

Lesson 11: The Graph of a Function

Student Outcomes

* Students understand set builder notation for the graph of a real-valued function: .
* Students learn techniques for graphing functions and relate the domain of a function to its graph.

Lesson Notes

The lesson continues to develop the notions of input and output from Lesson 9 and 10: if *f* is a function and is an element of its domain, then denotes the output of *f* corresponding to the input . In particular, this lesson is designed to make sense of the definition of the *graph of ,* which is quite different than the definition of the *graph of the equation* covered in the next lesson. The ultimate goal is to show that the “graph of ” and the “graph of ” both define the same set in the Cartesian plane. The argument that shows that these two sets are the same uses an idea that is very similar to the “vertical line test.”

Lessons 11 and 12 also address directly two concepts that are usually “swept under the rug” in K–12 mathematics and in ignoring, can create confusion about algebra in students’ minds. One is the universal quantifier “for all.” What does it mean and how can students develop concept images about it? (It is one of the two major universal quantifiers in mathematics—the other is “there exists.”) The other concept is the meaning of a variable symbol. Many wrong descriptions invoke the adage, “A variable is a quantity that varies.” A variable is neither a quantity, nor does it vary. As we saw in Module 1, it is merely a placeholder for a number from a specified set (the domain of the variable).

In this lesson, we build concept images of *for all* and *variable* using the idea of pseudo code—code that mimics computer programs. In doing so, we see *for all* as a for-next loop and *variables* as actual placeholders that are replaced with numbers in a systematic way. See the end of this lesson for more tips on Lessons 11 and 12.

A teacher knowledgeable in computer programming could easily turn the pseudo code in these lessons into actual computer programs. (The pseudo code was designed with the capabilities of Mathematica in mind, but any programming language would do.) ***Regardless, the pseudo code presented in this lesson is purposely designed to be particularly simple and easy to explain to students***. If the pseudo code is unduly challenging for students, consider "translating" the code for each example into set-builder notation either in advance or as a class. The first example has the following features:



**Pseudo code:**

**Specifies the domain of the variable of to be the set of integers.**

**Performs the instructions in the “loop body” first for equal to 1, then 2, then 3, then 4, then 5.**

**“loop body”**

**Substitutes the next element in the set for and runs the “loop body” instructions for that value of . For example, if the loop just completed for , “Next ” tells the computer to run the “loop body” instructions for .**

**Note about the lesson pace:** Every new example or exercise in this lesson usually inserts just 1 more line of pseudo code into the previous exercise or example. Use this continuity to your advantage in developing your lesson plans. For instance, you can just modify the example that is already on the board for each new example.

Classwork

Classwork

In Module 1, you graphed equations such as by plotting the points in the Cartesian plane by picking values and then using the equation to find the value for each value. The number of order pairs you plotted to get the general shape of the graph depended on the type of equation (linear, quadratic, etc.). The graph of the equation was then a representation of the solution set, which could be described using set notation.

In this lesson, we extend set notation slightly to describe the graph of a function. In doing so, we explain a way to think about set notation for the graph of a function that mimics the instructions a tablet or laptop might perform to “draw” a graph on its screen.

**Example 1 (5 minutes)**

Example 1

Computer programs are essentially instructions to computers on what to do when the user (you!) makes a request. For example, when you type a letter on your smart phone, the smart phone follows a specified set of instructions to draw that letter on the screen and record it in memory (as part of an email, for example). One of the simplest types of instructions a computer can perform is a *for-next loop*. Below is code for a program that prints the first 5 powers of 2:

|  |
| --- |
| Declare integer For all from 1 to 5  Print  Next |

The output of this program code is

* Go through the code with your students as if you and the class were a “computer.”

Here is a description of the instructions: First, is quantified as an integer, which means the variable can only take on integer values and cannot take on values like or . The “For” statement begins the loop, starting with . The instructions between “For” and “Next” are performed for the value , which in this case is just to “Print .” (Print means “print to the computer screen.”) Then the computer performs the instructions again for the next (), i.e., “Print ,” and so on until the computer performs the instructions for , i.e., “Print .”

Have students study the pseudo code and output and then ask the following:

* What is the domain of the variable ?
  + *Integers.*
* If is a function given by evaluating the expression for a number , what is the domain of the function given by the program?
  + *The set .*
* What is the range of ?
  + *The set .*

Point out to your students the similarity between mathematics and programming. In fact, it should come as no great surprise that many of the first computers and programming languages were invented by mathematicians. If you have time, feel free to discuss Blaise Pascal’s Calculator, Gottfried Leibniz’s Stepped Reckoner, and the life of George Boole (using articles found on *Wikipedia*).

Exercise 1 (3 minutes)

Exercise 1

Perform the instructions in the following programming code as if *you were a computer and your paper was the computer screen*:

|  |
| --- |
| Declare integer For all from 2 to 8  Print  Next |

Answer:

**Example 2 (4 minutes)**

Example 2

We can use almost the same code to build a set: first, we start with a set with 0 elements in it (called the *empty set*)and then increase the size of the set by appending one new element to it in each for-next step:

|  |
| --- |
| Declare integer Initialize as {} For all from 2 to 8  Append to   Print  Next |

Go through the code step-by-step with your students as if you and the class were a computer.

Note that is printed to the screen after each new number is appended. Thus, the output shows how the set “builds”:

Exercise 2 (4 minutes)

Exercise 2

We can also build a set by appending ordered pairs. Perform the instructions in the following programming code as if *you were a computer and your paper was the computer screen* (the first couple are done for you):

|  |
| --- |
| Declare integer Initialize as {} For all from 2 to 8  Append ) to  Next  Print |

Output:  
{, , \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_}

Answer: .

* Ask students why the set is only printed once and not multiple times like in the previous example.
  + *Answer: Because the “Print G” command comes after the for-next loop has completed.*

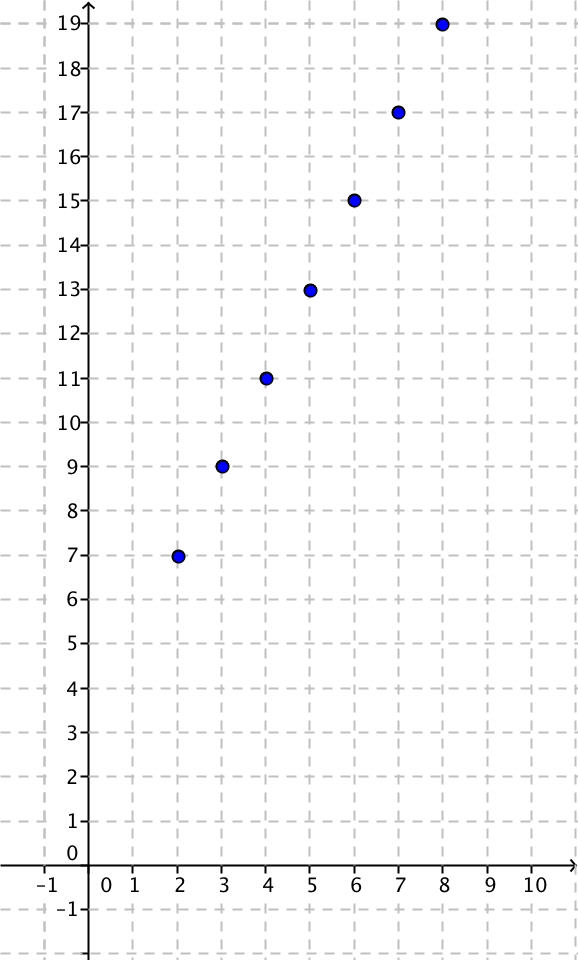
**Example 3 (4 minutes)**

Example 3

Instead of “Printing” the set to the screen, we can use another command, “Plot,” to plot the points on a Cartesian plane.

|  |
| --- |
| Declare integer Initialize as {} For all from 2 to 8  Append to  Next  Plot |

Output:



Some graphing calculators actually slow the computer down on purpose to give the human eye a sense of it plotting or drawing each point. If you have such a graphing calculator, graph an example with your students.

Make the point that, inside each for-next step, the variable has been replaced by a number at the beginning of the loop and that number does not change until all the instructions between the “For” and “Next” are completed for that step (“Next ” calls for a new number to be substituted into ).

In mathematics, the programming code above can be compactly written using set notation, as follows:

This set notation is an abbreviation for “The set of all points such that is an integer and .” Notice how the set of ordered pairs generated by the for-next code above,

also satisfies the requirements described by . It is for this reason that the set notation of the form

{type of element | condition on each element}

is sometimes called *set-builder notation*—because it can be thought of as building the set just like the for-next code.

If time permits, have students check that the set generated by the for-next instructions is exactly the same as the set described using the set-builder notation.

Discussion (7 minutes)

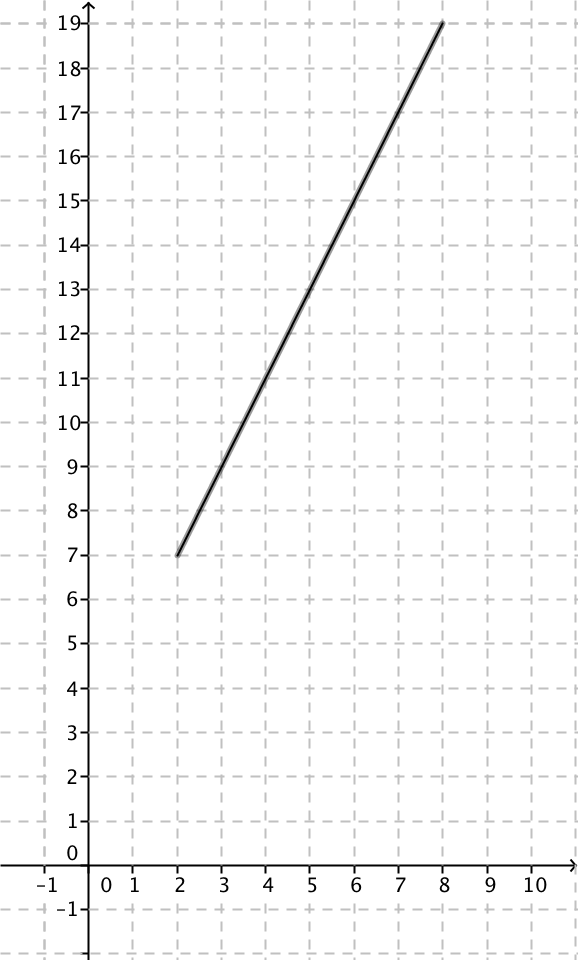
Discussion

We can now upgrade our notion of a for-next loop by doing a thought experiment: Imagine a for-next loop that steps through *all* real numbers in an interval (not just the integers). No computer can actually do this—computers can only do a finite number of calculations. But our human brains are far superior to that of any computer, and we can easily imagine what that might look like. Here is some sample code:

Students have misconceptions about how a graphing calculator generates a graph on its screen. Many students actually think that the computer is running through every real number, instead of the finite set of numbers needed to pixelate just the pixels needed to display the graph on the screen.

|  |
| --- |
| Declare real Let  Initialize as {} For all such that   Append ) to  Next  Plot |

The output of this thought code is the graph of for all real numbers in the interval :



Point out to students that a couple parts of the code have changed from the previous examples: (1) The variable is now quantified as a real number, not an integer. (2) For clarity, we named the function , where the function has domain and range . (3) The loop starts with the input value and appends to , and now we *imagine* that it steps one-by-one through every real number between 2 and 8, each time appending to the set . Finally, the loop finishes with appending to .

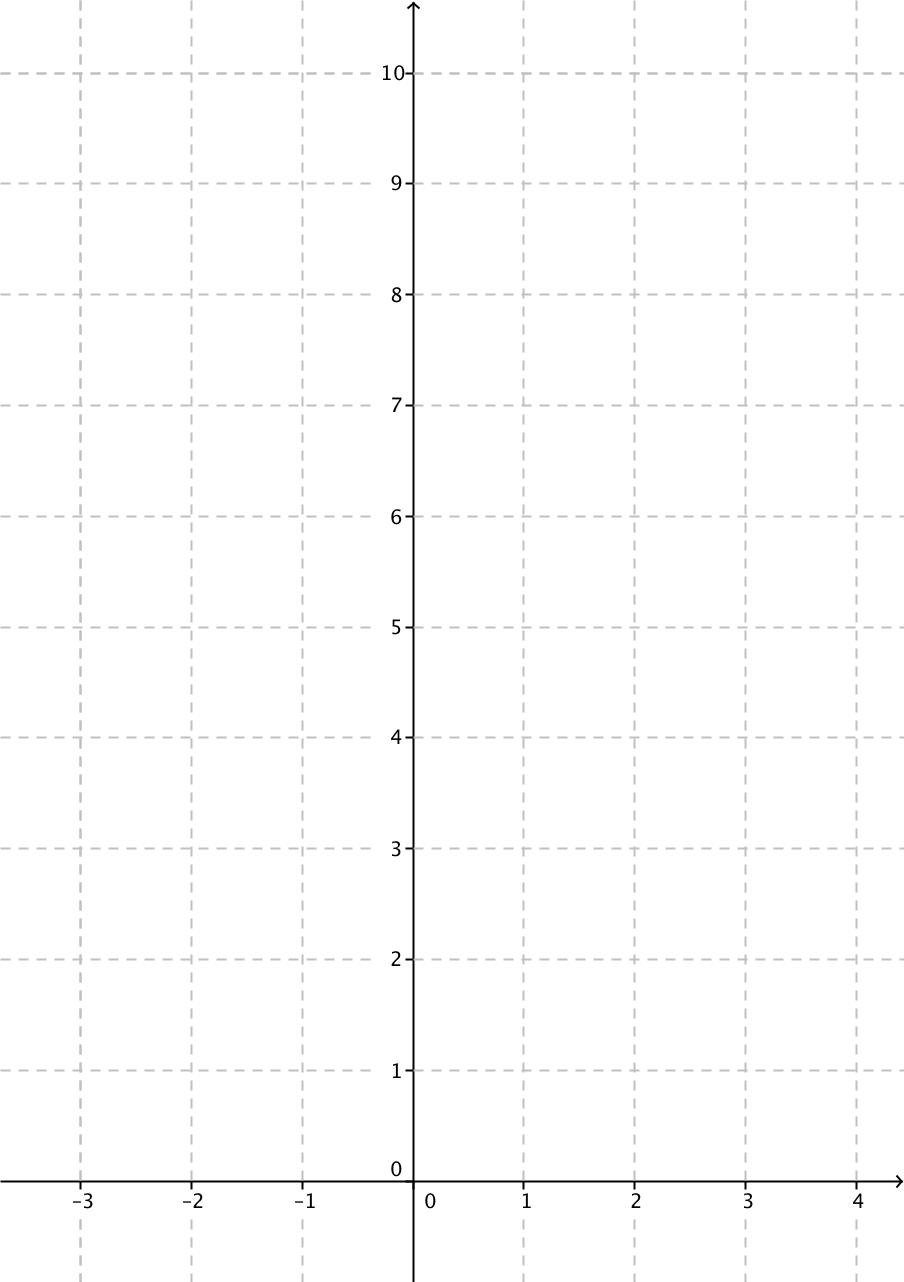
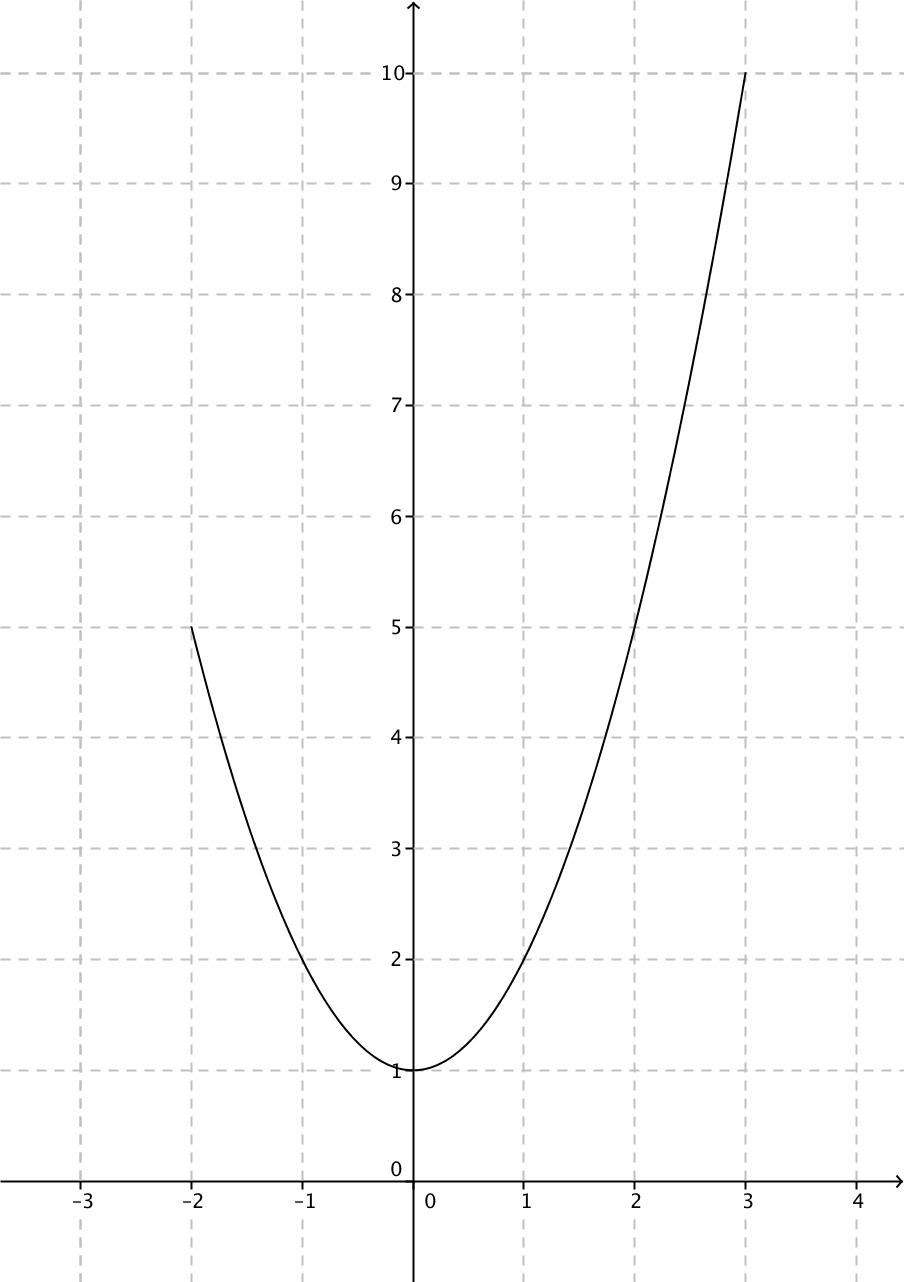
The resulting set , thought of as a geometric figure plotted in the Cartesian coordinate plane, is called the *graph of* . In this example, the graph of is a line segment.

Exercise 3 (10 minutes)

Exercise 3

* 1. Plot the function on the Cartesian plane using the following for-next thought code:

|  |
| --- |
| Declare real Let  Initialize as {} For all such that   Append to  Next  Plot |

Solution

Walk around the class and provide help on how to plot the function in the given domain. Remind students that the way we humans “draw” a graph is different than the way a computer draws a graph. We usually pick a few points (end points of the domain interval, the point (0, y-intercept), etc.), plot them first to get the general shape of the graph, and then “connect-the-dots” with an appropriate curve. Students have already done this for some time now but not in the context of Their biggest challenge will likely be working with function notation to get the -coordinate for a given -coordinate.

* 1. For each step of the for-next loop, what is the input value?

The number .

* 1. For each step of the for-next loop, what is the output value?

or the value of .

* 1. What is the domain of the function ?

The interval .

* 1. What is the range of the function ?

The interval for all in the domain.

* If time allows, ask students to describe in words how the thought code works.
  + *Answer: First, the domain of the variable is stated as the real numbers, the formula for is given, and the set is initialized with nothing in it. Then the for-next loop goes through each number between and inclusive and appends the point to the set . After every point is appended to , the graph of is plotted on the Cartesian plane.*

Closing (5 minutes)

Closing

The set built from the for-next thought code in Exercise 4 can also be compactly written in mathematics using set notation:

When this set is thought of as plotted in the Cartesian plane, it is the same graph. When you see this set notation in your homework and/or future studies, it is helpful to imagine this set-builder notation as describing a for-next loop.

In general, if is a function with domain , then its *graph* is the set of all ordered pairs,

thought of as a geometric figure in the Cartesian coordinate plane. (The symbol simply means “in.” The statement  
 is read, “ in .”)

Lesson Summary

Graph of : Given a function whose domain and the range are subsets of the real numbers, the graph of is the set of ordered pairs in the Cartesian plane given by

Exit Ticket (3 minutes)

**To the teacher: Further tips about this lesson and the next.** Lesson 11 and Lesson 12 use programming code to develop distinctly *mathematical concepts*. In particular:

* The lessons are designed to give a conceptual image to the meaning “for all,” which is a concept that is often used but rarely (if at all) explained.
* The lessons are designed to help students develop a conceptual image of a variable as a placeholder—that students have complete control over what can be substituted into the placeholder (just as the programmer does when he/she instructs the computer to call the “next ”).
* The pseudo code in Lesson 11 and the pseudo code in Lesson 12 help students to understand that the “graph of ” and the “graph of ” are differently generated sets. The first set is generated with just a straight for-next loop, and the second set is generated by a nested for-next loop that tests every point in the plane to see if it is a solution to . It is study of how the two “computer programs” are different that helps students see that the way the sets are generated is different.
* The pseudo code in Lesson 12 for generating the graph of is also another way for students to envision how the points in the graph of any equation in two variables can be generated (like ). In particular, it explains why we set “” equal to .” The notation can appear strange to students at first: students might wonder, “In eighth grade we said that was a function. Now, in ninth grade, we say is a function, so what is so special about ? Doesn’t that just mean ?” The pseudo code helps students see that is a name for a function and that is an equation in the sense of Module 1.
* It is through the study of the two types of programs that the two differently generated graphs can be shown to be the same set; that is, the graph of is the same set as the graph of . The critical issue that helps equate the two sets is a discussion about the definition of function and why the definition guarantees that there is only one -value for each -value. The pseudo code is designed to help you make this point.
* Without saying so, Lesson 11 suggests how graphs are created when students use their graphing calculators. (The “Plot” function is just another for-next loop that pixelates certain pixels on the screen to form the graph.) The lessons are designed to help demystify these “little black boxes” and to plant the seed in your students’ heads that programming computers may not be as hard as they thought.
* Finally, the long division algorithm is the first nontrivial algorithm students learn. Up to this point in their education, it is also one of the only algorithms they have learned. The pseudo code in these two lessons gives students a chance to see other types of useful algorithms that are also easy to understand.

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Lesson 11: The Graph of a Function**

**Exit Ticket**

1. Perform the instructions for the following programming code as if *you were a computer and your paper was the computer screen*.

|  |
| --- |
| Declare integer Let  Initialize as {} For all from 3 to 2  Append to  Next  Plot |

1. Write three or four sentences describing in words how the thought code works.

**Exit Ticket Sample Solutions**

1. Perform the instructions for the following programming code as if *you were a computer and your paper was the computer screen*.



|  |
| --- |
| Declare integer Let  Initialize as {} For all from 3 to 2  Append to  Next  Plot |

1. Write three or four sentences describing in words how the thought code works.

Answer: The first three lines declare the domain of the variable to be the integers, specifies the formula for , and sets to be the empty set with no points in it. Then the for-next loop goes through each integer between 3 and 2 inclusive and appends the point to the set . After every point is appended to , the graph of is plotted on the Cartesian plane.

Problem Set Sample Solutions

1. Perform the instructions for each of the following programming codes as if *you were a computer and your paper was the computer screen*.

|  |
| --- |
| Declare integer For all from 0 to 4  Print  Next |

Answer:

|  |
| --- |
| Declare integer For all from 0 to 10  Print  Next |

Answer:

|  |
| --- |
| Declare integer For all from 2 to 8  Print  Next |

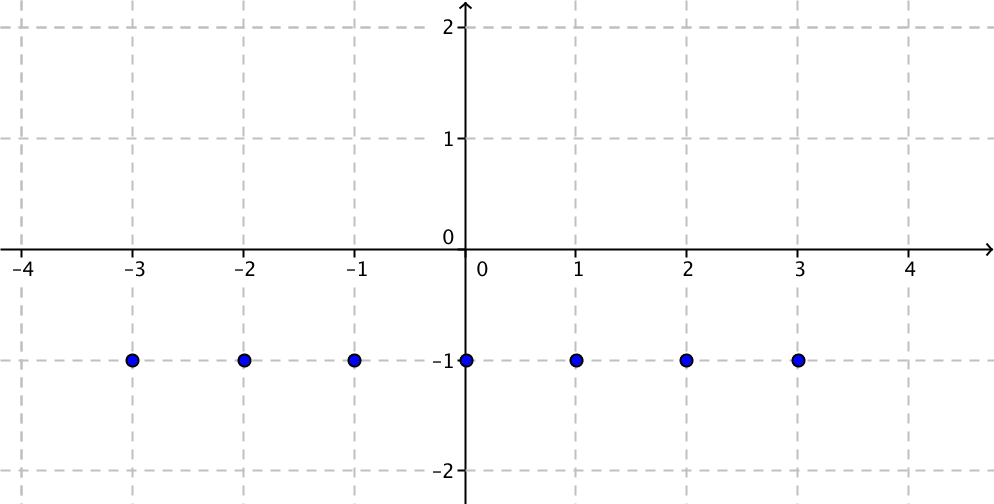
Answer:

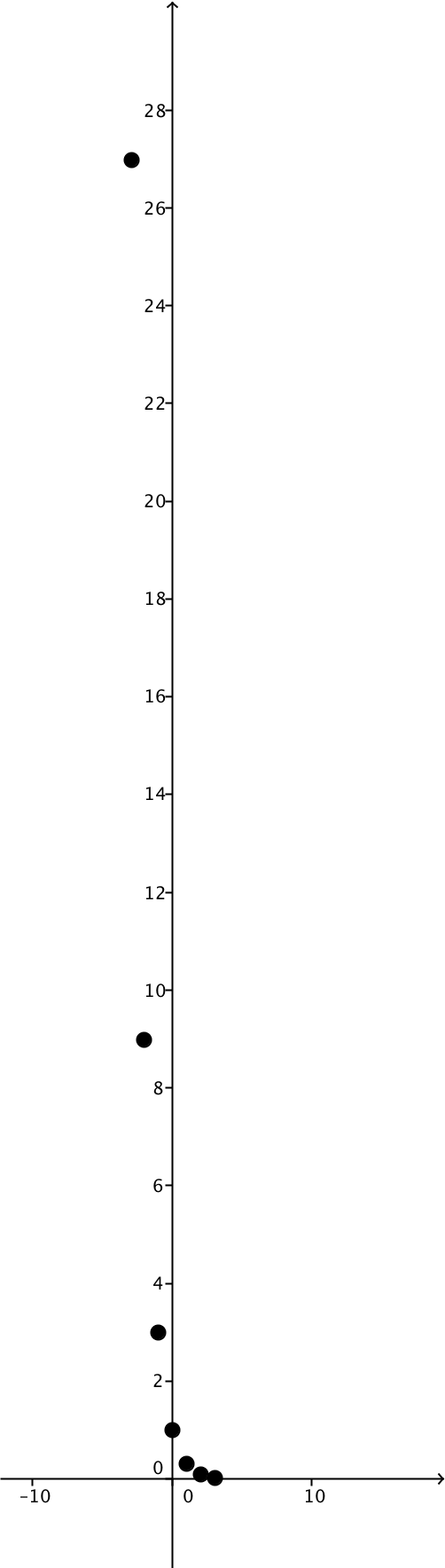
|  |
| --- |
| Declare integer For all from 0 to 4  Print  Next |

Answer:

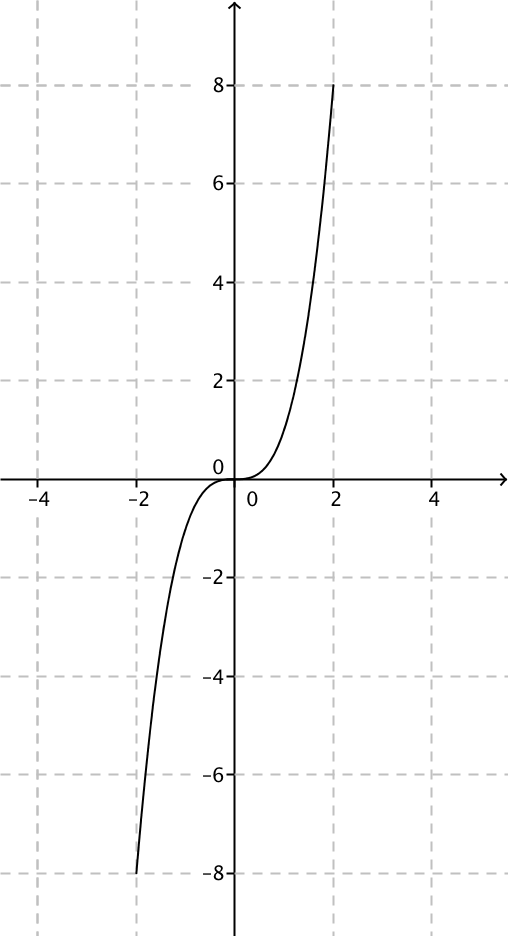
1. Perform the instructions for each of the following programming codes as if *you were a computer and your paper was the computer screen*.

|  |
| --- |
| Declare integer Let  Initialize as {} For all from 3 to 3  Append to  Next  Plot |



* 1. 

|  |
| --- |
| Declare integer Let  Initialize as {} For all from 3 to 3  Append to  Next  Plot |

* 1. 

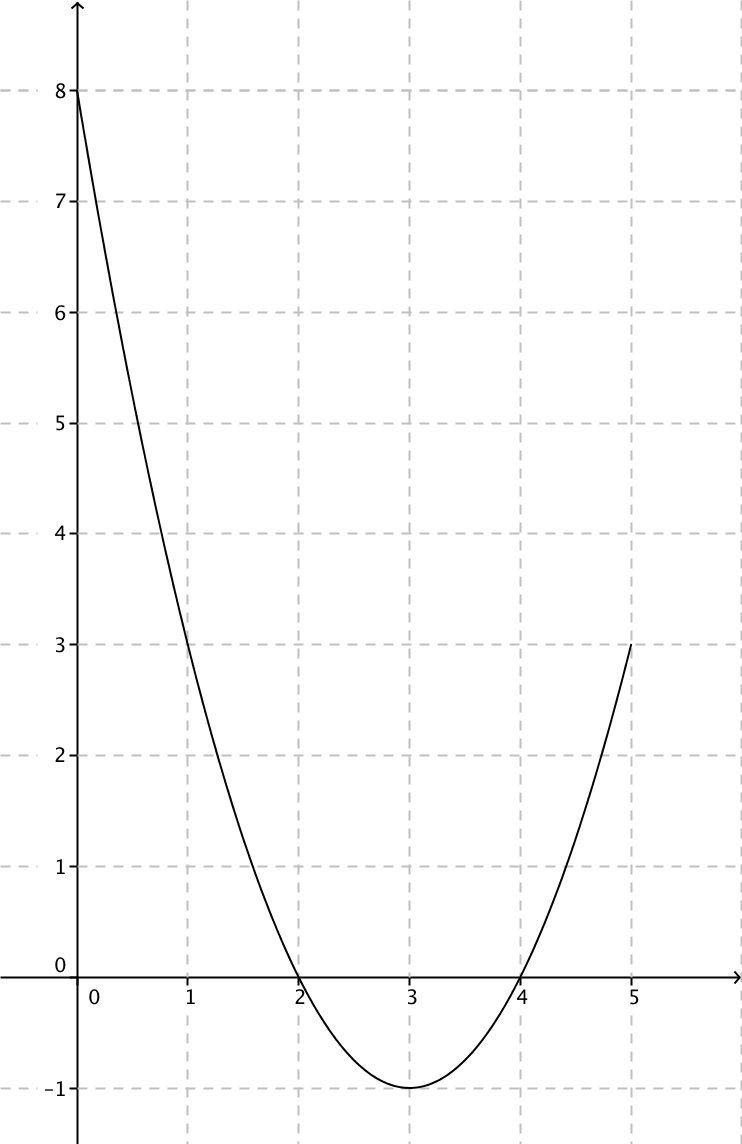
|  |
| --- |
| Declare real Let  Initialize as {} For all such that   Append to  Next  Plot |

1. Answer the following questions about the “thought code:”

|  |
| --- |
| Declare real Let  Initialize as {} For all such that   Append to  Next  Plot |

* 1. What is the domain of the function ?

Answer: .

* 1. Plot the graph of according to the instructions in the thought code.  
       
     
  2. Look at your graph of . What is the range of ?

Answer: for all in the domain.

* 1. Write three or four sentences describing in words how the thought code works.

Answer: First, the domain of the variable is stated as the real numbers, the formula for is given, and the set is initialized with nothing in it. Then the for-next loop goes through each number between and inclusive and appends the point to the set . After every point is appended to , the graph of is plotted on the Cartesian plane.

1. Sketch the graph of the functions defined by the following formulas, and write the graph of as a set using set-builder notation. (Hint: Assume the domain is all real numbers unless specified in the problem.)



Graph of =



Graph of =



Graph of =



Graph of =



Graph of =

* 1. ,



Graph of = or Graph of =

* 1. ,

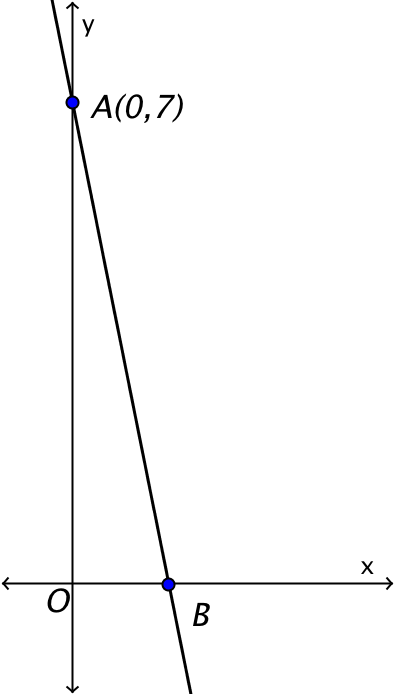


Graph of = or Graph of =

* 1. ,



Graph of = or   
Graph of =

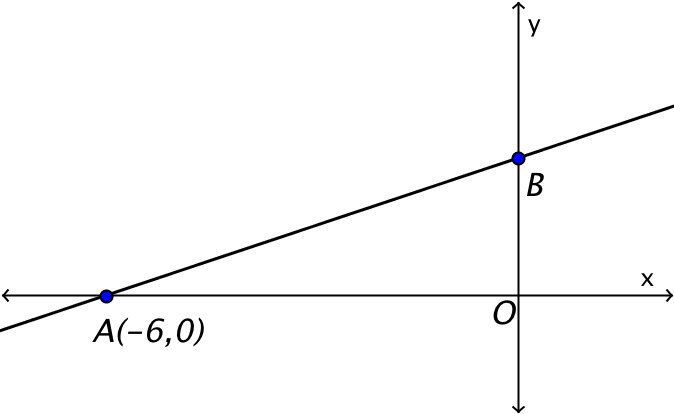
1. The figure shows the graph of .  
     
      
   1. Find the value of .

Answer: .

* 1. If the graph of intersects the -axis at , find the coordinates of .

Answer: .

1. The figure shows the graph of .



* 1. Find the value of .

Answer: .

* 1. If the graph of intersects the -axis at , find the coordinates of .

Answer:

* 1. Find the area of triangle .

Answer: 9 square units.